ANALISIS PERFORMA CAPACITY SCHEDULING ALGORITHM PADA SISTEM JOB SCHEDULING HADOOP

CAPACITY SCHEDULING ALGORITHM PERFORMANCE ANALYSIS ON HADOOP JOB SCHEDULING SYSEM

[1]Alfian Dzulfikar Khabibi [2] Gandeva Bayu Satrya, S.T., M.T. [3] Anton Herutomo, ST., M.Eng.
[1 2 3] Fakultas Informatika, Universitas Telkom, Bandung
[1]alfiandzulfikark@gmail.com, [2]gandeva.bayu.s@gmail.com, [3]anton.herutomo @gmail.com

Abstrak

Hadoop merupakan *framework* software berbasis *java* dan *open-source* yang berfungsi untuk mengolah data yang besar secara terdistibusi dan berjalan diatas *cluster* yag terdiri dari beberapa gabungan komputer. Arsitektur Hadoop terdiri dari 2 *layer* pokok, yaitu *layer MapReduce* dan layer Hadoop *Distributed File System* (HDFS). Map Reduce adalah komponen service kunci yang berfungsi untuk melakukan proses komputasi *Big Data* secara paralel dan terdistribusi dan (HDFS) berfungsi untuk menyediakan *bandwidth* sangat tinggi yang di agregasi ke semua *cluster (node).*

Dalam *MapReduce* terdapat terdapat job scheduler yang berfungsi untuk memetakan antrian *job* yang masuk. *Job scheduler* default dari Hadoop adalah FIFO dan Hadoop mengizinkan penggantian *job scheduler* default dengan *custom job scheduler*. *Capacity Scheduling* merupakan *job scheduler* pada Hadoop yang berkarakteristik memberikan *capacity guarantee* kepada antrian yang masuk pada *queue* yang telah disediakan sehingga bisa ditekannya nilai Fail Rate. Tetapi karena *resource* harus dibagi menjadi beberapa bagian maka performansi *Response Time* dan *Job Troughput* menurun. Pada algoritma FIFO nilai maksimal *Job Fail rate* yaitu 10%, sedangkan pada *Capacity Scheduling* nilai *Job Troughput* maksimal adalah 4,3%.

Kata kunci : Hadoop, data, *multi-user, Capacity Scheduling,* FIFO

Abstract

Hadoop is a software framework based on Java and open-source that serves to process large data terdistibusi and running on the cluster that rested on several computers combined. Hadoop architecture consists of two basic layers, namely layer and layer MapReduce Hadoop Distributed File System (HDFS). Map Reduce is a key service component that functions to perform processes in parallel computing and Big Data distributed and (HDFS) serves to provide very high bandwidth in aggregation to all clusters (nodes).

In MapReduce there are job scheduler that serves to map the incoming job queue. Job scheduler default of FIFO and Hadoop Hadoop is a job scheduler to allow replacement of default with a custom job scheduler. Capacity Scheduling is a job scheduler on Hadoop which characterized provide capacity guarantee to incoming queues in the queue that has been provided so that it can repress the value Fail Rate. But because the resource should be divided into several sections, the performance of Response Time and Job Throughput decreased. In the FIFO algorithm maximum value Job Fail rate is 10%, whereas in Job Throughput Capacity Scheduling maximum value is 4.3%

***Key words**: Hadoop, data, multi-user, Capacity Scheduling, FIFO*

I Pendahuluan

1.1 Latar Belakang

Sebagai pengguna, tentu kita menginginkan layanan perpindahan data cepat, tepat, dan efisien. Maka platform Hadoop dapat menjadi suatu jawaban karena Hadoop sudah cukup terkenal dan *open- source*. Hadoop merupakan implementasi *open-source* dari *MapReduce* yang menggunakan *job scheduler* sebagai memetakan tugas[8]. *MapReduce* harus menghindari transmisi data yang tidak perlu dengan cara meningkatkan lokalitas data agar performasinya bagus. Hadoop, sebuah *open-source framework* untuk pemrosesan *data-sets* skala besar dalam *clusters hardware* komputer yang terjangkau dan mudah diperoleh. Pada umumnya *framework* Hadoop dikembangkan dalam bahasa Java, dengan beberapa *source code* dalam bahasa C dan command line utilities ditulis sebagai shell-scripts[4].

Capacity scheduling merupakan *job scheduler* algoritma penjadwalan yang hampir optimal. *Capacity scheduling* dibuat untuk memaksimalkan *throughput* yang diterima oleh user dengan memberi keyakinan bahwa tiap klien yang mengakses akan mendapatkan resource[2]. Implementasi Hadoop dengan *capacity scheduling* juga digunakan oleh Yahoo[1]. Cara kerja *Capacity scheduling* yaitu dengan membuat *sub-queue*, menyediakan

control, dan prediksi. Dengan cara diatas semua klien mendapatkan *capacity guarantee* karena Semua *queue* akan dialokasikan pada sebagian kecil *grid* dan akan mendapatkan akses pada *source* yang telah dialokasikan.

Tugas akhir ini akan menggunakan FIFO dan *Capacity Scheduling* dalam Hadoop *job scheduler*. Penggunaan ini akan dibandingkan dengan implementasinya pada berbagai macam *job* dengan menggunakan parameter *Job Fail Rate, Job Troughput*, dan *Response Time* sebagai acuan perhitungan performansi sistem. Serta perbedaan performa *job scheduler* FIFO dan *Capacity Scheduling* pada berbagai macam *job* dan berbagai jumlah *job* pada masing-masing *job scheduler*.

2 Dasar Teori

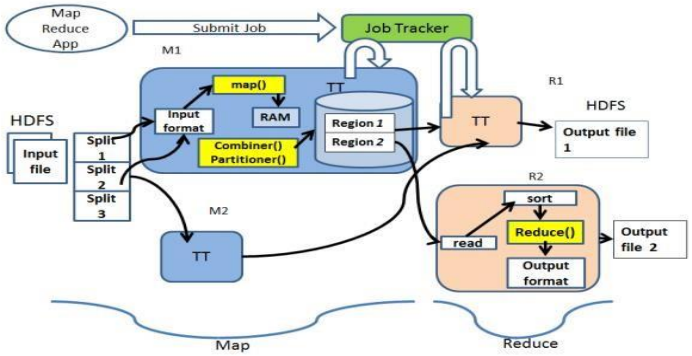
2.1 Hadoop

Hadoop adalah *open-source framework* untuk pemrosesan *data-sets* skala besar dalam *clusters hardware* komputer yang terjangkau dan mudah diperoleh. Hadoop adalah file system yang digunakan oleh berbagai situs informasi bersekala besar seperti google.com [6].

Hadoop adalah sebuah *open-source framework* yang mengimplementasikan *MapReduce pararel programing model*. Hadoop terdiri dari 2 komponen utama, yaitu *MapReduce* dan *user-level file system* yang berguna untuk mengatur penggunaan storage resource dantar beberapa *cluster*. Secara jelas sebagai berikut :

a. *MapReduce Engine*
Terdiri dari *master job-tracker* dan satu *TaskTracker per cluster-node*. *Master job-tracker* berguna untuk mengatur scheduling dan monitoring berbagai komponen dari job task dalam slave dan mengeksekusi kembali task yang gagal dieksekusi sebelumnya.

b. *HDFS : The Hadoop Distributed File System*
Menyediakan akses global dalam *cluster*. HDFS diimplementasikan dalam 2 servis, *NameNode* dan *DataNode*. *NameNode* berguna untuk mengatur HDFS *directory tree* dan sentralisasi servis dalam *cluster* sedangkan *DataNode* berguna menyimpan data dalam blok-blok. [8].



Gambar Error! No text of specified style in document.**-1 Hadoop HDFS Sistem**

Pada umumnya *framework* Hadoop dikembangkan dalam bahasa *Java*, dengan beberapa *source code* dalam bahasa C dan *command line utilities* ditulis sebagai *shell-scripts*[4]. Beberapa ecosistem hadoop yang dapat digunakan antara lain :

a. *Apache HDFS*
Menyediakan *service* penyimpanan yang besar dan *multipe machines*. HDFS terinspirasi dari Google File System. Menyediakan *bandwith* yang sangat besar yang dapat diakomodasi oleh user.

b. Hadoop *MapReduce*
model pemograman untuk pengolahan data skala besar.

c. *Hadoop YARN*
sebuah platform manajemen sumber daya yang bertanggung jawab atas pengelolaan sumber daya komputasi dalam sebuah *cluster* dan digunakan untuk penjadwalan aplikasi pengguna.

d. *Red Hat GlusterFS*
Gluster File System adalah *Scale-out-network File System* yang dikembangkan oleh *Gluster* pada tahun 2011. Pada tahun 2012 disupport oleh *Red Hat* yang bernama *Red Hat Storage Server*.

e. *Quantcash File System*
Sebuah *package software* yang berdasarkan *MapReduce* atau *batch-processing workloads*. Hampir sama dengan HDFS atau bisa disebut alternatif dari HDFS yang dikembangkan dalam bahasa C++ dan *fixed-footprint memory management*.

f. *Ceph*
Sebua *software* gratis yang didesain untuk memetakan *obejct*, blok, dan file memori pada *singgle cluster* komputer.

g. Lutstre File System
High-performance file terdistribusi yang digunakan pada skala luas. Digunakan untuk meremote dapa pada SAN (*Storage area Network*).

2.2 Job Scheduling

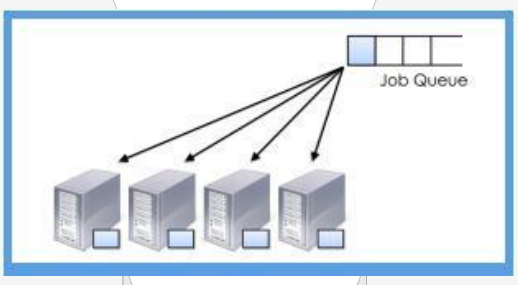
Job scheduling dalam Hadoop adalah cara penyelesaian dari pencarian data sehingga mendapat hasil yang diharapkan. Pada Hadoop, sistem algoritma *job scheduling* menggunakan FIFO scheduling. Karena Hadoop *open-source* maka, algoritma tersebut dapat dimodifikasi sesuai keinginan pengembang, untuk hasil yang lebih baik.[10]

2.3 Multi-user

Multi-user adalah istilah dalam sistem operasi atau perangkat lunak aplikasi yang memperbolehkan akses oleh beberapa pengguna dalam waktu bersamaan ke sistem operasi atau aplikasi tersebut. Istilah lawannya yaitu *single-user* mengacu kepada suatu sistem operasi yang hanya bisa digunakan oleh satu pengguna setiap saat. Peranan dalam masalah ini adalah, jika *user* yang mengakses *server* terlalu banyak maka mengakibatkan kerja yang ditanggung oleh *server* bertambah berat, sebaliknya jika dilihat dari sisi *user*, kualitas data menurun.

2.4 FIFO Scheduling

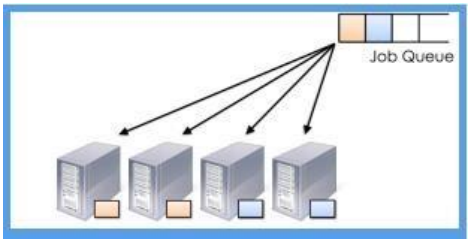
Algoritma FIFO merupakan *default* yang digunakan oleh Hadoop pada proses penjadwalan. Algoritma ini mengatasi permasalahan antrian pada *job* dengan cara menjalakan sebuah *job* yang datang untuk pertama kali. Algoritma FIFO tidak menangani adanya skala prioritas dan perhitungan *long jobs* atau *short jobs*. Sehingga mengakibatkan penggunaan algoritma FIFO kurang efektif. Pada implementasi algoritma FIFO dalam server berskala besar, algoritma FIFO dapat menurunkan performansi dari sisi server terutama pada sistem yang mempunyai layanan *sharing data* pada *multiple-user*[5]



Gambar Error! No text of specified style in document.**-2 Ilustrasi FIFO**

2.5 Capacity scheduling

Capacity scheduling adalah sebuah algoritma *MapReduce* yang menyediakan *resource pool*, *queuing priority*, dan *pool leave accesss-control*. Beberapa *queue* memiliki prioritas tersendiri untuk mengakses *resource* dalam *cluster*. Ketika *cluster* mempunyai *slot task* yang tidak terpakai, maka *queue* dapat menggunakan slot yang sedang berstatus *idle*, meskipun *queue* tersebut tidak memiliki prioritas. Kemudian jika datang *job* dengan *queue priority* datang atau diaktifkan, maka queue yang memiliki *priorty* tersebut langsung mendapatkan slot dari slot yang dipakai oleh *job* yang tidak mempunyai *priority*, dengan menghentikan layanan terhadap *job* yang tidak memiliki *priority* tadi[7][9].



Gambar Error! No text of specified style in document.**-3 Ilustrasi Capacity Scheduling**

Capacity Scheduling menangani *cluster* yang besar dengan memberikan *capacity guarantee*. Ide sederhana dari *Capacity Scheduling* adalah *resource* yang ada pada *cluster* dibagi kepada beberapa antrian. *Capacity*

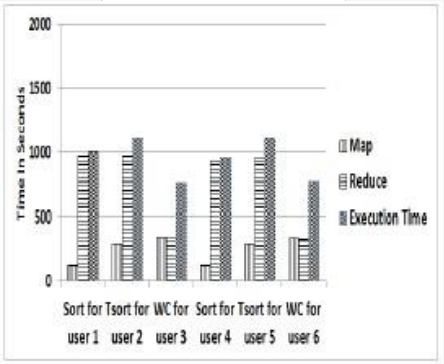
Scheduling menyediakan limit untuk mengatur stabiliti dan data yang diterima oleh user. Beberapa fitur yang diberikan oleh capacity scheduling :

a. *Hierarchical Queues* - Membagi antrian dan menyediakan resource yang akan dpakai oleh sub queues dan, menyediakan kontrol dan prediksi.
b. *Capacity Guarantees* – Semua antrian akan dijamin mendapatkan akses data seperti yang dialokasikan.
c. *Security* – Setiap antrian mendapatkan antrian ACL yanga akan mengontrol user dan memasukkan aplikasi pada singgle queue. Terdapat fitur yang memungkinkan user tidak dapat memodif aplikasi dari user lain.
d. *Elasticity – Resource* bebas yang dapat dialokasikan pada semua antrian lebih dari kapasitas antrian tersebut. Ketika antrian sedang dikerjakan pada capacity yag tidak memenuhi pada future point time, ketika antrian tersebut selesai digunakan, kemudian akan dibentuk suatu prediksi untuk menyediakan queue selanjutnya.
e. *Multu-tenancy-comprehensive* mengeset limit untuk menghindari aplikasi, user, atau queue singgle untuk memonopoli resource yang ada.
f. *Resource-based scheduling – support resource-intensive,* yaitu ketika aplikasi dapat memilih resource yang lebih dari *default*.[2]

Setiap *queue* yang telah didefinisikan oleh *cluster administration* harus mempunyai *configuration block* yang terdapat pada *hadoop-site.xml*. Dari semua *job* yang telah diproses oleh cluster dapat dipantau melauli *jobtracker*. *Jobtracker* akan menampilkan beberapa informasi tentang proses *scheduling*. *Jobtracker* adalah fitur dari Hadoop yang digunakan untuk memantau pengeksekusian *job queue*.

Dalam *Capacity scheduler* terdapat beberapa parameter yang menetukan hasil dari pengeksekusian setiap *job* berdasarkan resource cluster seperti berapa antrian yang dikonfigurasi untuk user. Beberapa parameter yang ada dalam *Capacity Scheduler* antara lain:

1. *NumberofQueue* dan *QueueCapacity*
   QueueCapacity adalah capacity guarantee yang didapat masing-masing *job* yang dimasukkan dalam antrian. Waktu lama eksekusi dipengaruhi oleh banyaknya *numberofqueue* dan pengurangan *queuecapacity* karena reosurce dibagi-bagi pada *disk* dan *bandwidth* jaringan.



Gambar Error! No text of specified style in document.**-4 2 queue dengan 50% capacity**

2. *MaximumCapacity*
   Parameter ini mengizinkan sebuah *queue* untuk menggunakan *capacity* yang tak terpakai pada queue yang lain yang tidak terpakai. Sebuah *queue* dapat menggunakan resource pada cluster diantara value *queuecapacity* dan maximumcapacity (100% *default*). Seperti yang terlihat pada gambar 2-4 dan 2-4 setiap ditambah queue dan konfigurasi *maximum capacity* akan menambah waktu *response time*.
3. *Minimumuserlimitpercent*
   Parameter ini mengizinkan limit yang dialokasikan pada setiap user yang menginpukan job terhadap server. Seiring dengan dibaginya minimumuserlimitpercent, maka akan semakin lama waktu eksekusi karena beberapa job akan dikerjakan bersamaan.
4. Userlimitfactor
   Parameter ini mengizinkan sau user untuk mengambil lebih banyak slot yang telah dikonfigurasikan pada queuecapacity. Tetapi parameter ini tidak relevant karena tidak ada pengaruh yang besar pada setiap job yang dieksekusi.
5. *Support-priority*

Parameter ini mengizinkan untuk memberikan *priority* pada sebuah *queue* dalam *capacity*.[8].

2.6 Hadoop *Multi-Node Cluster*

Hadoop multi-node cluster merupakan gabungan dari beberapa *server* Hadoop, yaitu 1 *server master* dan beberapa *slave server*. Beberapa *server* ini dikonfigurasi melalui file *mapred-site.xml, core-site.xml*, dan *hdfs-site.xml*. *Server master* dapat bekerja sebagai *slave server* dan *slave server* hanya bekerja sebagai *slave server* saja.

2.7 Job

Job merupakan suatu pekerjaan yang dapat dikerjakan oleh sistem Hadoop. *Job* ini dapat diberikan oleh *user* yang terhubung dengan *server* unruk keperluan tertentu. *Job* pada Hadoop akan dibagi menjadi 2 bagian yaitu *Map* dan *Reduce*[10].

Map merupakan bagian awal dari suatu job untuk membangun sutau informasi yang terdapat pada suatu data menjadi data yang siap diproses sesuai dari pekerjaan yang akan diproses oleh *server*. Sedangkan *reduce* adalah suatu proses yang dikerjakan oleh *server* untuk mengerjakan proses sesuai dengan pekerjaan yang diberikan oleh *user* atau *server* itu sendiri pada suatu data yang telah melalui proses *Map*[10].

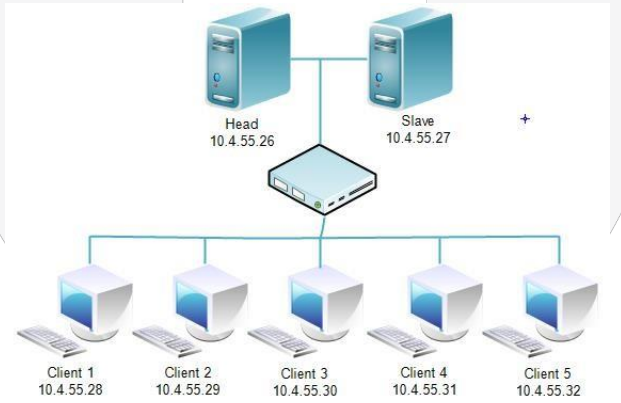
2.8 Parameter Pengujian

Parameter yang dipakai dalam pengujian ini yaitu *Job fail Rate, Response Time, dan Job Troughput*. Rincian paameter yang digunakan sebagai berikut:

1. *Job fail Rate*
   Pada parameter ini akan diukur jumlah *job* yang gagal dan telah diulang kembali pekerjaannya atau yang benar benar gagal dan merupakan kesalahan dari *resource* dari *server* hadoop[3].
2. *Response Time*
   Parameter ini akan diukur dari jumlah waktu yang dibutuhkan oleh *server* untuk mengerjakan semua *job* yang masuk pada satu antrian[5]. Satuan yang dipakai adalah menit.
3. *Job Troughput*
   Pada parameter ini akan diukur jumlah *job* yang berhasil diselesaikan dalam selang waktu tertentu. Nilai yang dihasilkan akn diperoleh dari awal suatu *job* diselesaikan pada suatu antrian[3]. Satuan pada parameter ini adalah *job*/menit.

3 Perancangan dan Implementasi

Perancangan sistem tugas akhir ini difokuskan pada performansi dari *job scheduling* yang diterapkan pada sistem *cluster* server Hadoop. Parameter yang digunakan sebagai pembanding performansi dari job scheduling adalah *Job Fail Rate*, *Job Troughput*, dan *Response Time*. Penjelasan konfigurasi jaringan, perangkat keras, dan perangkat lunak yang digunakan dalam tugas akhir ini akan dijelaskan dibawah ini:



Gambar Error! No text of specified style in document.**-5 Gambar Topologi**

Pada Gambar diatas, server Hadoop terdiri dari 2 buah komputer yang akan disatukan menjadi sebuah *cluster*. Kedua server tersebut berperan sebagai master dan slave kemudian akan dikonigurasi menjadi Hadoop *multi-node cluster*, dimana head server bekerja sebagai master dan slave sedangkan slave server berfungsi sebagai slave. Pada rancangan topologi jaringan ini, Hadoop head server memiliki IP (Internet Protocol) 10.4.55.26/24 dan slave server memiliki IP (Internet Protocol) 10.4.55.27/24 . Client terdiri dari 5 buah komputer yang digunakan sebagai pengirim *job* pada server. Komputer clien dapat mengirim lebih dari satu *job*. *Job* yang dikirimkan oleh client ada tiga jenis yaitu, *job wordcount* yang bekerja mencari kata yang mirip dalam sebuah resource *plaintext*, job grep yaitu menghitung kata yang telah didefinisikan oleh user dalam sebuah resource *plaintext*, dan *randomtextwriter* yang bekerja menuliskan 10 GB random data pada DFS Map/Reduce, setiap map yang dikirimkan akan menjadi file *input* dan menulis *random byte* dan fase reduce tidak digunakan.

4 Pengujian dan Analisis Hasil Implementasi

Skenario pada pengujian Tugas Akhir ini dibagi menjadi 7 skenario dengan algoritma FIFO dan *Capacity Scheduling*. Berikut tabel skenario yang akan dijalankan pada pengujian Tugas Akhir ini, sebagai berikut:

Tabel 4-1 Skenario Pengujian

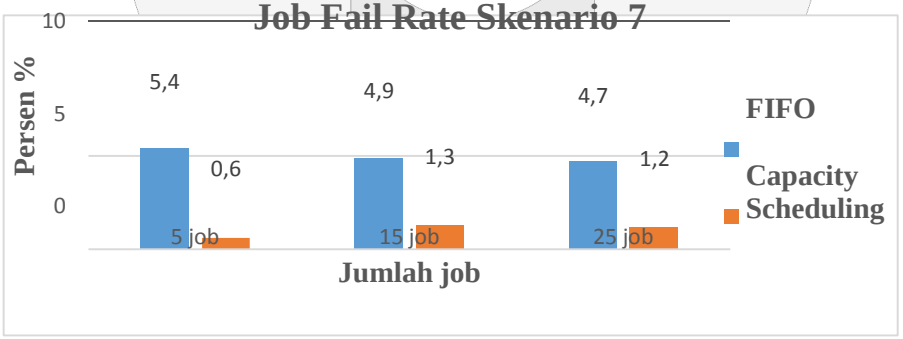
No	FIFO			Capacity Scheduling		
Skenario 1	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs
Skenario 2	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs
Skenario 3	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs
Skenario 4	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs
Skenario 5	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs
Skenario 6	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs
Skenario 7	5 jobs	10 jobs	25 jobs	5 jobs	10 jobs	25 jobs

Pada tabel 3 – 2 diatas skenario 1,2, dan 3 menggunakan masing-masing satu jenis *job* yaitu *job wordcount, job grep*, dan *job randomtextwriter*. Kemudian skenario 4,5,d an 6 menggunakan kombinasi dari 2 kombinasi jenis job diatas yaitu job *wordcount* dan *job grep, job wordcount* dan *job randomtextwriter*, dan *job grep* dan *job randomtextwriter*. Skenario terakhir menggunakan kombinasi dari tiga jenis *job*, yaitu *job wordcout, job grep*, dan *job randomtextwriter*.

Semua skenario tersebut akan dikirim melalui lima *client* yang akan mengirimkan *job* pada server. Jumlah *job* yang dikirimkan tiap *client* tidak harus sama, karena jumlah user tidak mempengaruhi performansi dari *multi-cluster* Hadoop yang diuji, melainkan antrian dan jumlah *job* yang masuk pada server

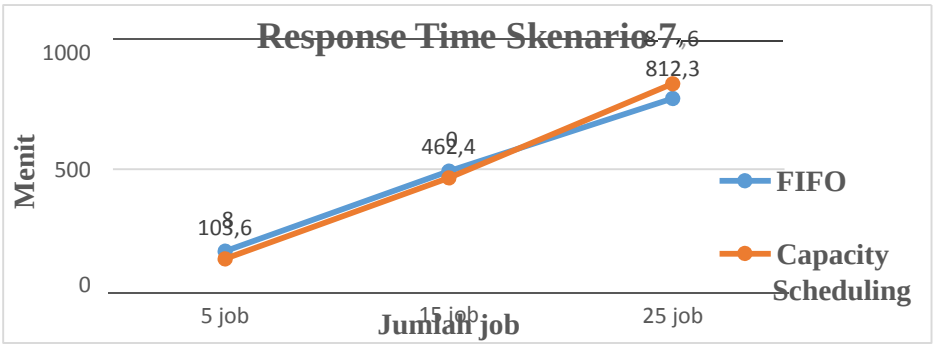
4.1 Analisis menggunakan kombinasi 3 jenis job

Pada skenario ini akan dilakukan pengujian terhadap tiga jenis *job*, yaitu *job wordcount, job grep* dan *job randomtextwriter* pada algoritma FIFO dan *Capacity Scheduling*. Dengan karakteristik *Capacity Scheduling* yaang memberi *capacity guarantee* maka menghasilkan *job Fail Rate* yang rendah pada *Capacity Scheduling*. Resource file yang diakses berukuran 2,4 GB, 743 MB, dan 20 MB dengan total 48 *maps.* Pada job *randomtextwriter* akan dieksekusi dengan menuliskan data random sebesar 10 GB dengan total 20 *maps.* Skenario ini bercirikan dengan job yang bersifat *homogen* dan mempunyai *resource* data yang beragam dengan jumlah 5 *jobs*, 15 *jobs*, dan 25 *jobs.*



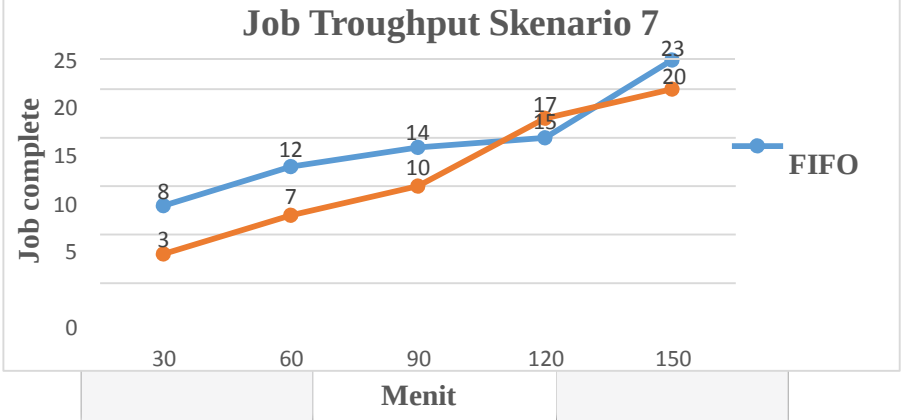
Gambar Error! No text of specified style in document.**-6 Graik *Job Fail Rate* Skenario 7**

Berdsarkan grafik diatas parameter *Job Fail Rate* pada algoritma FIFO berkisar antara 4%-5% pada setiap penambahan *jobs* yang dilakukan. Sedangkan pada *Capacity Scheduling* parameter *Job Fail Rate* berkisar pada angka 1% pada setiap penambahan job yang dilakukan. Hal ini merupakan karakteristik dari *Job Scheduler Capacity Sheduling* yang memberikan resource guarantee kepada antrian yang masuk terhadap *queue* yag telah didefinisikan yang mengakibatkan bisa ditekannya nilai *Fail* pada *job* yang dimasukkan yang terlihat pada lampiran C. Kombinasi *job randomtextwriter* pada skenario ini mengakibatkan nilai *Fail Rate* maksimal 1,3% pada *Capacity Scheduling*, karena *job randomtextwriter* merupakan job yang cukup besar untuk dikerjakan.



Gambar Error! No text of specified style in document.**-7 Grafik *Response Time* Skenario 7**

Pada grafik diatas parameter *Response Time* pada algoritma FIFO memiliki waktu yang lebih singkat dari pada *Capacity Scheduling*. Pada algoritma *Capacity Scheduling* memiliki ciri dimana terjadi penambahan jumlah *Response Time* tiapa interval *job* yang dimasukkan pada antrian. Hal ini merupakan ciri dari Capacity Scheduling dimana resource yang ada dibagi menjadi beberapa *queue* sehingga performansi pengeksekusian *job* menjadi semakin lama seperti pada lampiran semakin bertambahnya *job* maka semakin lama *Capacity Scheduling* mengeksekusi *job* sehingga total *Response Time* meningkat. Penggunaan kombinasi *job randomtextwriter* juga mempengaruhi peningkatan *Response Time* tiap *job* seperti pada lampiran C.



Gambar Error! No text of specified style in document.**-8 Grafik *Job Troughput* Skenario 7**

Pada Grafik diatas sangat terlihat sekali perbedaan dari algoritma FIFO dan *Capacity scheduling*. Terliat perbedaan *Job Troughput* yang sangat besar dari 2 *scheduler* tersebut. Pada algoritma FIFO dapat mengeluarkan 23 job pada 150 menit sedangkan pada *Capacity Scheduling* pada 150 mengeluarkan troughput 20 job. Hal ini terjadi karena terdapat kombinasi *job randomtextwriter* yang berkarakteristik *job* yang cukup lama untuk diekrjakan karena menuliskan data random sebesar 10GB yang mana *Capacity scheduling* kurang maksimal karena *resource* dari *cluster* harus dibagi-bagi, sehingga pengerjaan antara satu *job* dengan *job* yang lain terjadi perbedaan waktu eksekusi, sehinga berpengaruh pada *job troughput*.

5 Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan tujuan serta hasil pengujian dan analisis data yang telah dilakukan pada penggunaan algoritma FIFO dan *Capacity Scheduling* sebagai *job scheduler* Hadoop *MapReduce*, maka dapat ditarik kesimpulan sebagai berikut:

1. *Capacity Scheduling* memiliki performansi efektif daripada FIFO pada jenis job grep dengan penurunan *Job Fail Rate* menjadi 1,1% pada total 25 *job* dan peningkatan *Job Troughput* sebesar 4 *job* pada 22,5 menit.
2. Pada algoritma FIFO nilai maksimal *Job Fail rate* yaitu 10% pada skenario 3, sedangkan pada *Capacity Scheduling* nilai *Job Trougput* maksimal adalah 4,3% pada skenario 3. *Capacity Scheduling* memberikan *capacity guarantee* yang mengakibatkan dapat ditekannya nilai *Job Fail Rate*, tetapi karena *resource* yang ada dibagi menjadi beberapa *sub queue*, performansi *Response Time* dan *Job Troughput* menurun.
3. *Capacity Scheduling* lebih efektif digunakan untuk mengerjakan jenis *job* yang memiliki *maps* dan ukuran yang kecil seperti *job grep*. Jika digunakan untuk mengeksekusi job yang relatif besar seperti *job*

wordcount dan *randomtextwriter*, performansi *Capacity Scheduling* akan semakin menurun tiap penambahan job.

5.2 Saran

Pengembangan lebih lanjut dari tugas akhir ini dapat dilakukan dengan menambah server Hadoop. Menambahkan jumlah *jobs*, menggunakan kombinasi berbagai jenis *job* lain, dan penggantian algoritma *job scheduler* Hadoop lain pada server Hadoop.

Daftar Pustaka

[1] He Chen, Ying Lu, David Swanson, Matchmaking: *A New MapReduce Scheduling Technique*, Nebraska

[2]http://hadoop.apache.org/docs/r2.5.0/hadoop-yarn/hadoop-yarn-site/CapacityScheduler.html diakses tanggal 23 Oktober 2014

[3] XIA Yang, Lei WANG, Qiang ZHAO, Gongxuan ZHANG, *Research on Job Scheduling Algorithm in Hadoop*, China, 2011

[4] Alex Holmes, *Hadoop in Practice*, USA, 2009

[5] Rasooli Aysan, Douglas G. Down, *A Hybrid Scheduling Approach for Scalable Heterogeneous Hadoop Systems*, Canada

[6] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, *Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling*

[7] Jason Vaner, *Pro Hadoop Build scalable, distributed application in the cloud*

[8] Jagmohan Chauhan, Dwight Makaroff and Winfried Grassmann, *The Impact of Capacity scheduling Configuration Settings on MapReduce Jobs*

[9] B.Thirumala Rao, Dr. L.S.S.Reddy, *Survey on Improved Scheduling in Hadoop MapReduce in Cloud Environments*

[10] Tom White, Hdoop : *The Definitve Guide*, United State of America, 2009